| NODIS Library | Program Formulation(7000s) | Search |

NASA Procedural Requirements

COMPLIANCE IS MANDATORY

NPR 7150.2A

Effective Date:
November 19, 2009
Expiration Date:
November 19, 2014

Printable Format (PDF)

Request Notification of Change (NASA Only)

Subject: NASA Software Engineering Requirements

Responsible Office: Office of the Chief Engineer

| TOC | Preface | Chapter1 | Chapter2 | Chapter3 | Chapter4 | Chapter5 | Chapter6 | AppendixA | AppendixB | AppendixC | AppendixD | AppendixE | ALL |

Appendix A. Definitions

A.1 **Accuracy**. The difference between a parameter or variable (or a set of parameters or variables) within a model, simulation, or experiment and the true value or the assumed true value (Definition from source document: NASA-STD-7009, Standard for Models and Simulations.).

A.2 **Accredit**. The official acceptance of a software development tool, model, or simulation, (including associated data) to use for a specific purpose.

A.3 **Analysis**. The post-processing or interpretation of the individual values, arrays, files of data, or execution information.

A.4 **Contracted Software**. Software created for a project by a contractor or subcontractor.

A.5 **Data**. Information for computer processing (e.g., numbers, text, images, and sounds in a form that is suitable for storage in or processing by a computer).

A.6 **Deviation**. A documented authorization releasing a program or project from meeting a requirement before the requirement is put under configuration control at the level the requirement will be implemented.

A.7 **Embedded Software**. Software that is part of a larger system and performs some of the requirements of that system (Definition from source document: ISO/IEC 24765:2009 Systems and Software Engineering Vocabulary.).

A.8 **Establish and Maintain**. The responsible project, organization, or individual must formulate, document, use/deploy, and keep current the object (usually a document,

requirement, process, or policy).

A.9 **Glueware**. Software created to connect the off-the-shelf software/reused software with the rest of the system. It may take the form of "adapters" that modify interfaces or add missing functionality, "firewalls" that isolate the off-the-shelf software, or "wrappers" that check inputs and outputs to the off-the-shelf software and may modify to prevent failures.

A.10 **Government Off-The-Shelf (GOTS) Software**. This refers to Government-created software, usually from another project. The software was not created by the current developers (see software reuse). Usually, source code is included and documentation, including test and analysis results, is available. That is, the government is responsible for the GOTS software to be incorporated into another system. (Definition from source document: NASA-GB-8719.13, NASA Software Safety Guidebook.)

A.11 **Insight**. Surveillance mode requiring the monitoring of customer-identified metrics and contracted milestones. Insight is a continuum that can range from low intensity, such as reviewing quarterly reports, to high intensity, such as performing surveys and reviews. (Definitions from source document: NPR 8735.2, Management of Government Safety and Mission Quality Assurance Surveillance Functions for NASA Contracts.)

A.12 **Independent Verification and Validation (IV&V)**. Verification and validation performed by an organization that is technically, managerially, and financially independent of the development organization (ISO/IEC 24765:2008 systems and software engineering vocabulary).

A.13 **Legacy/Heritage**. Software products (architecture, code, requirements) written specifically for one project and then, without prior planning during its initial development, found to be useful on other projects. See software reuse.

A.14 **Major Engineering/Research Facility**. Used in this document to show research, development, test, or simulation facilities representing a significant NASA investment which contains software that supports programs and projects managed under NPR 7120.5, NPR 7120.7, or NPR 7120.8. Examples include: high-fidelity, motion-base flight simulator facilities (e.g., Vertical Motion Simulator at Ames), wind tunnels (e.g., National Transonic Facility at LaRC), vacuum chambers (e.g., Space Power Facility at GRC/Plum Brook), air traffic control facilities (e.g., North Texas Research Station), and engine test stands (e.g., J-2X test stand at Stennis Space Center). Related major facility designations are contained in NPR 8800.15, Section 3.8.1, Designation of Major Facilities.

A.15 **Mathematical Model**. The mathematical equations, boundary values, initial conditions, and modeling data needed to describe the conceptual model (ASME V&V 10). (Definition from source document: NASA-STD-7009, Standard for Models and Simulations.)

A.16 **Mission Critical**. Item or function that must retain its operational capability to assure no mission failure (i.e., for mission success). (Definition from source document: NPR 8715.3, NASA General Safety Manual Program Requirements.)

A.17 **Model**. A description or representation of a system, entity, phenomena, or process. (Definition from source document: NASA-STD-7009, Standard for Models and Simulations.) Only for the purpose of this document, the term "model" refers to only those models which are implemented in software.

A.18 **Modified Off-The-Shelf (MOTS) Software**. When COTS, legacy/heritage software is reused, or heritage software is changed, the product is considered "modified." The changes can include all or part of the software products and may involve additions, deletions, and specific alterations. An argument can be made that any alterations to the code and/or design of an off-the-shelf software component constitutes "modification," but the common usage allows for some percentage of change before the off-the-shelf software is declared to be MOTS software. This may include the changes to the application shell and/or glueware to add or protect against certain features and not to the off-the-shelf software system code directly. See off-the-shelf.

A.19 **Off-The-Shelf Software**. Software not developed in-house or by a contractor for the specific project now underway. The software is generally developed for a different purpose than the current project.

A.20 **Operational Software**. Software that has been accepted and deployed, delivered to its customer, or is deployed in its intended environment.

A.21 **Primary Mission Objectives**. Outcomes expected to be accomplished which are closely associated with the reason the mission was proposed, funded, developed, and operated (e.g., objectives related to top-level requirements or their flow down).

A.22 **Process Asset Library**. A collection of process asset holdings that can be used by an organization or project. (Definition from source document: CMMI® for- Systems Engineering/Software Engineering/Integrated Product and Process Development Supplier Sourcing.)

A.23 **Program**. A strategic investment by a Mission Directorate or Mission Support Office that has a defined architecture and/or technical approach, requirements, funding level, and a management structure that initiates and directs one or more projects. A program defines a strategic direction that the Agency has identified as critical.

A.24 **Project**. A specific investment having defined goals, objectives, requirements, life-cycle cost, a beginning, and an end. A project yields new or revised products or services that directly address NASA's strategic needs. They may be performed wholly in-house; by Government, industry, academia partnerships, or through contracts with private industry.

A.25 **Reuse**. See software reuse.

A.26 **Risk Management**. An organized, systematic decision-making process that efficiently identifies, analyzes, plans, tracks, controls, communicates, and documents risk to increase the likelihood of achieving program/project goals. (Definition from source document: NPR 8735.2, Management of Government Quality Safety and Mission Assurance Surveillance Functions for NASA Contracts.) Risk management includes Risk-Informed Decision Making and Continuous Risk Management in an integrated framework. This is done to foster proactive risk management, to better inform decision making through better use of risk information, and then to more effectively manage implementation risks by focusing the Continuous Risk Management process on the baseline performance requirements emerging from the Risk-Informed Decision-Making process. (Definition from source document: NPR 8000.4, Agency Risk Management Procedural Requirements.)

A.27 **Scripts**. A sequence of automated computer commands embedded in a program that tells the program to execute a specific procedure (e.g., files with monitoring, logic, or

commands used by software to automate a process or procedure).

A.28 **Sensitivity Analysis**. The study of how the variation in the output of a model can be apportioned to different sources of variation in the model input and parameters. (Definition from source document: NASA-STD-7009, Standard for Models and Simulations.)

A.29 **Simulation**. The imitation of the characteristics of a system, entity, phenomena, or process using a computational model. (Definition from source document: NASA-STD-7009, Standard for Models and Simulations.) Only for the purpose of this document, the term "simulation" refers to only those simulations which are implemented in software.

A.30 **Software**. Computer programs, procedures, scripts, rules, and associated documentation and data pertaining to the development and operation of a computer system. Software includes programs and data. This also includes COTS, GOTS, MOTS, reused software, auto generated code, embedded software, firmware, and open source software components.

A.31 **Software Architecture**. The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the properties of those components, and the relationships between them. The term also refers to documentation of a system's software architecture. Documenting software architecture facilitates communication between stakeholders, documents early decisions about high-level design, and allows reuse of design components and patterns between projects.

A.32 **Safety-Critical Software**. See definition in NASA-STD-8719.13, Software Safety Standard.

A.33 **Software Engineering**. The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software: that is, the application of engineering to software. (Definition from source document: IEEE 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology.)

A.34 **Software Peer Review/Inspection**. A visual examination of a software product to detect and identify software anomalies, including errors and deviations from standards and specifications (IEEE 1028-2008 IEEE Standard for Software Reviews and Audits). Guidelines for software peer reviews/inspections are contained in NASA-STD-2202-93, NASA Software Formal Inspection Standard.

A.35 **Software Reuse**. A software product developed for one use but having other uses or one developed specifically to be usable on multiple projects or in multiple roles on one project. Examples include, but are not limited to, COTS products, acquirer-furnished software products, software products in reuse libraries, and pre-existing developer software products. Each use may include all or part of the software product and may involve its modification. This term can be applied to any software product (such as, requirements and architectures), not just to software code itself. Often this is software previously written by an in-house development team and used on a different project. GOTS software would come under this category if the product is supplied from one Government project to another Government project. (Definition from source document: NASA-GB-8719.13, NASA Software Safety Guidebook.)

A.36 **Software Validation**. Confirmation that the product, as provided (or as it will be

provided), fulfills its intended use. In other words, validation ensures that "you built the right thing."

A.37 **Software Verification**. Confirmation that work products properly reflect the requirements specified for them. In other words, verification ensures that "you built it right."

A.38 **Static Analysis**. The process of evaluating a system or component based on its form, structure, content, or documentation. (Definition from source document: ISO/IEC 24765:2008 systems and software engineering vocabulary.)

A.39 **Subsystem**. A secondary or subordinate system within a larger system. (Definition from source document: ISO/IEC 24765:2008 systems and software engineering vocabulary.)

A.40 **System**. The combination of elements that function together to produce the capability required to meet a need. The elements include hardware, software, equipment, facilities, personnel, processes, and procedures needed for this purpose. (Definition from source document: NPR 7123.1A NASA Systems Engineering Processes and Requirements.)

A.41 **Uncertainty**. (1) The estimated amount or percentage by which an observed or calculated value may differ from the true value. (2) A broad and general term used to describe an imperfect state of knowledge or a variability resulting from a variety of factors including, but not limited to, lack of knowledge, applicability of information, physical variation, randomness or stochastic behavior, indeterminacy, judgment, and approximation (adapted from NPR 8715.3B, NASA General Safety Program Requirements).

A.42 **Waiver**. A documented authorization intentionally releasing a program or project from meeting a requirement after the requirement is put under configuration control at the level the requirement will be implemented.

A.43 **Wrapper**. See glueware definition.

DISTRIBUTION:
NODIS